本源司南：高效利用量子资源的量子操作系统

孔伟成[1,2],王俊超[2,4],韩永健[2],吴玉椿[2],张 昱[3],窦猛汉[1],方 圆[1],郭国平[1,2]

[1](合肥本源量子计算科技有限责任公司,安徽 合肥 230026)

[2](中国科学院量子信息重点实验室（中国科学技术大学）,安徽 合肥 230026)

[3](中国科学技术大学 计算机科学与技术学院,安徽 合肥 230027)

[4](数学工程与先进计算国家重点实验室,河南 郑州 450002)

通讯作者: 郭国平，E-mail: gpguo@ustc.edu.cn

摘 要：随着超导等量子处理器技术的不断进步，高效利用量子处理器、实现量子计算机与经典计算平台的有效协作，已成为量子计算实用化发展的迫切需求。本源司南就是一种面向此需求的量子操作系统，它提供量子任务调度、量子资源管理、量子程序编译、量子比特自动化校准等服务，统一、高效地管理量子计算资源。本文提出用于本源司南的多量子处理器负载均衡、基于量子线路映射的多量子程序并行计算、基于隐式马尔科夫链的量子比特自动化校准等技术方案，通过将本源悟源等量子计算平台接入本源司南前后的数据对比：在测试 4 种代表性的量子线路（QFT,GHZ,DJ,BV）实验中相较于 BMT 算法的映射效果，经过本源司南映射后线路的保真度至少提高 10%;在运行 GHZ 量子线路实验中，本源司南提供的单量子处理器并行计算与多量子处理器负载均衡计算能力使量子处理器的运行效率至少提高 120%。本源司南能协同调度量子处理器（QPU）、经典计算机等计算资源，为量子计算的广泛使用提供了有效的资源管理。

关键词：本源司南；量子操作系统；量子计算；量子处理器

Origin Pilot: A Quantum Operating System for Efficient Usage of Quantum Resources

KONG Wei-Cheng [1,2], WANG Jun-Chao[2,4], HAN Yong-Jian[2], WU Yu-Chun[2], ZHANG Yu[3], DOU Meng-Han[1], FANG Yuan[1], GUO Guo-Ping[1,2]

[1](Origin Quantum Computing Company Limited, Hefei 230026, China)

[2](CAS Key Laboratory of Quantum Information (University of Science and Technology of China), Hefei 230026, China)

[3](School of Computer Science and Technology, University of Science and Technology of China, Hefei 230027, China)

[4](State Key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou 450002,China)

Abstract: With the continuous progress of quantum processor technologies, to efficiently use of quantum processors and collaboration between quantum processor and classical computing have become an urgent requirement for further application of quantum computing. The Origin Pilot is a quantum operating system oriented to this requirement. It provides services, such as quantum task scheduling, quantum resource management, quantum program compilation, and automatic calibration of quantum bit, and manages quantum computing resources efficiently and unified. In this paper, we propose multiple quantum processor load balancing, multiple quantum program parallel computing based on quantum line mapping, quantum bit automatic calibration based on implicit Markov chain. By comparing before and after using the Origin Pilot on OriginQ Wuyuan quantum computing platforms, we found that the fidelity is increased by at least 10% (in the experiment of four typical quantum circuits (QFT, GHZ,DJ,BV), compared with the mapping effect of BMT algorithm)and the operating efficiency is increased by at least 120% (In the experiment of running GHZ quantum circuit, the parallel computing ability of single quantum processor and load balancing computing ability of multiple quantum processor provided by Origin Pilot) with Origin Pilot. Origin Pilot cooperates quantum processor (QPU), classical computer and other computing resources, which provides an effective resource management for quantum computing.

Key words: Origin Pilot; Quantum Operating System; Quantum Computing; Quantum Processor

1 概述

量子计算是量子力学与计算科学相结合的一种新型计算方式[1] 。它以量子比特为基本计算单元，利用量子叠加（量子纠缠）、量子测量等量子特性，在特定问题上可实现相对于经典计算的指数加速[2] [3] 。它在非对称密码的破译[4]、量子化学[5] 、金融[6] 、机器学习[7] [8] 等诸多领域相对于经典计算有计算优势。

现阶段，实现量子计算的物理体系主要包括超导系统[9] 、半导体自旋系统[10] 、离子阱系统[11] [12] 、光学系统[13] [14] 等。随着这些系统中硬件（如：超导系统中的材料与工艺的改善[15] 、环境噪声优化[16] 、控制技术与电子学系统发展[17] [18] ）以及软件（如，量子控制架构[19] 和基础量子软件[20] ）的完善，量子计算的初步应用[21] [22] 已成为可能。一方面，Google 于 2019 年首次在 53 位量子比特的 Sycamore 芯片上验证了量子计算在随机线路采样问题中相较于经典超级计算机所具有的更强大的计算能力[23] ，实现了"量子优越性" [24] 。另一方面，基于云端的量子计算（提供商包括：IBM、D-Wave、谷歌、Rigetti、本源量子等）已经开始服务于不同的任务，并积累了一批非量子计算专业用户与研究人员。

然而，随着量子计算资源、量子计算需求以及普通用户的迅速增加，如何管理量子计算设备[25] 以及高效使用量子计算资源[26] 等问题逐渐成为量子计算应用的关键障碍。为此，Henry Corrigan-Gibbs 等人首次提出了 Quantum Operating System 的概念，描述了三种潜在的量子计算机底层硬件架构，分别面向量子 FPGA（附加在经典 CPU 上的现场可编程逻辑门阵列）、量子 x86（量子 x86 实现了经典 x86 指令集，以及通用量子指令集）系统、量子分布式系统[27] 。Reid Honan 等人通过分析不同量子程序之间的关系、量子比特映射以及比特的连通性，构造了一个可串行的冲突图(Conflict Graph)，根据此图提出了一种多量子程序在单量子处理器中并行执行的方法[28] 。为了让量子应用程序开发人员只关注量子应用本身，而无需了解底层量子硬件（超导、离子阱等）的物理细节，英国的 Riverlane 公司发布了量子操作系统 Deltaflow.OS[29] 。而奥地利的 ParityQC 公司针对特定量子处理器的量子程序编译优化发布了量子操作系统 ParityOS。尽管这上述量子操作系统在各自的方向都提出了解决方案，但是在量子资源使用方面，依然存在多量子处理器的调度和量子比特自动校准及优化等问题亟需解决。

多量子处理器的调度：当前基于云服务的量子计算机，量子程序使用的量子处理器是由用户自行选择设置，而非系统根据量子处理器的可用性、可满足性进行任务分配。这导致局部量子资源排队拥堵，而其他量子资源闲置浪费。为提高效率，需要实现对多个量子程序按其消耗的计算资源（简称量子资源）在多个量子处理器中进行自动的资源分配与调度。

自动校准并优化量子比特质量：量子比特易受环境及噪声影响，并导致其性能不稳定，进而降低量子逻辑门的保真度，最终导致量子线路的运算结果不正确。及时校准是保持量子比特性能的主要手段，目前主要采用离线定时校准。显然，这样的校准方式有以下两个问题：a)两次校准间隔期间，量子比特质量无法保证；b)校准过程中，质量良好的比特无法提供计算服务，造成计算资源的浪费。有鉴于此，谷歌实现了一套名为 Optimus 的自动化校准系统[23] ，实现了按需校准，但尚不清楚其是否实现了在线校准。

针对以上挑战，本文提出能高效利用量子资源的量子操作系统——本源司南。本源司南通过提供的量子线程调度、量子比特自动校准优化、量子程序编译、量子资源管理四个核心服务，有效地解决上述问题，提高了量子计算资源的使用效率。

本文主体结构如下：第二节主要介绍了量子计算、量子操作系统的基本概念；第三节介绍本源司南量子操作系统的总体架构及工作流程；第四节给出了多量子处理器负载均衡、多量子程序并行计算（同步并行）和量子比特自动化校准等关键问题的解决方案；第五节，本源司南的实验验证内容；第六节，介绍本源司南的展望及未来工作。

2 预备知识

2.1 量子计算

量子比特

量子比特是量子计算的基本单元。在经典计算体系中，一个比特在同一个时间只能表示 0 或者 1，但量子比特可以表示为 0 和 1 的叠加状态(Superposition State)，即$|\psi>=\alpha|0>+\beta|1>$,其中$\alpha$，$\beta\in\mathbb{C}$，$|\alpha|^2+$

$|\beta|^2 = 1$。

量子测量

量子测量是获取量子态信息的手段。量子测量时量子比特按一定的概率坍缩到$|0>$或$|1>$上。

量子逻辑门

量子逻辑门是作用于量子比特上的酉变换，它实现量子比特状态的变换。与经典逻辑门不同，量子逻辑门是可逆门。为实现普适的量子计算，我们只需要实现单量子比特酉变换和两比特 CNOT 门即可，特别的，单量子比特门可以由 Hadamard 门、T 门和 S 门生成。常见的量子逻辑门在计算基$|0>$和$|1>$下可表示如下：

单比特量子逻辑门	多比特量子逻辑门
$H = \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{pmatrix}$	$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$
$S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$	$CZ = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$
$T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}$	$SWAP = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$
$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$	$CU = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & U_{00} & U_{01} \\ 0 & 0 & U_{10} & U_{11} \end{pmatrix}$
$Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$	$iSWAP = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & cos\left(\frac{\theta}{2}\right) & i\,sin\left(\frac{\theta}{2}\right) & 0 \\ 0 & i\,sin\left(\frac{\theta}{2}\right) & cos\left(\frac{\theta}{2}\right) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$
$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$	$CR = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{i\theta} \end{pmatrix}$
$U_3 = \begin{pmatrix} cos\left(\frac{\theta}{2}\right) & -e^{i\lambda} \times sin\left(\frac{\theta}{2}\right) \\ e^{i\phi} \times sin\left(\frac{\theta}{2}\right) & e^{i\lambda + i\phi} \times cos\left(\frac{\theta}{2}\right) \end{pmatrix}$	$Toffoli = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$

量子线路

量子线路模型是最常用的量子计算模型，在此模型中，任意酉变换都可以通过普适量子逻辑门的有序组合来实现，而这个量子逻辑门序列的有序组合我们称之为量子线路（与经典门线路类似，仅需要将经典逻辑门换为量子逻辑门）。量子线路可以可视化为量子线路图，如图 1 所示：

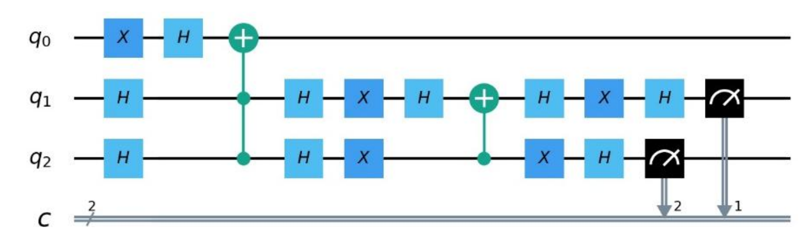


图 1　Grover 算法量子线路示意图

一般的，在量子计算中，量子线路的输入量子态为$|0>^{\otimes n}$，而在量子线路结束时，伴随着量子测量来获得计算的结果。

量子程序

量子程序是由量子逻辑门、量子测量、经典计算等操作组成的有序序列，它可以包括多段量子线路。

量子态保真度

由于环境噪声及量子处理器自身品质的影响[30]，实际量子处理器执行结果往往与理想情况下经过量子门操作得到的结果有一定的偏差[31] [32] 。这种偏差可以用理想量子态和实际量子态之间的保真度来衡量：保真度越大，偏差越小，系统的计算结果就越好；反之就越差。

2.2 经典计算机与量子计算机的区别

由于量子计算与经典计算在信息处理的基本单元上遵从不同的物理规律（前者遵从量子力学，后者遵从经典力学），这导致二者在计算机体系架构的实现以及计算方式上都有本质不同。特别是：

- **量子比特映射**

量子处理器中，由于工艺不同, 即使同一物理体系也可能有不同的拓扑结构，且它们支持不同的基础逻辑门。因此，同一个量子程序在编译时，不同的量子比特映射方案的效率可能有巨大的差异。特别的，每个量子比特的实时状态不同，这为量子比特映射带来了额外复杂度。这是量子比特与经典比特在调用时的根本差异。

- **量子并行计算**

线程是经典操作系统能够进行运算调度的基本执行单元，单核 CPU 在同一时刻只能执行一个线程的指令。多个线程可以在经典操作系统中实现线程切换[33]，进而实现在单核 CPU 下的多线程并发计算[34] 。

然而，在量子计算中，量子程序是量子处理器的最小执行单元。由于量子计算本身的特征，多个量子程序之间不能实现像经典程序中断切换类似的操作。但多个量子程序使用不同物理量子比特时，多个量子程序在同一个量子处理器上可实现并行计算。

- **量子比特自动化校准**

在经典计算系统中，其芯片工艺成熟，比特性能稳定，不会在短时间内出现浮动。因此，在进行资源优化时，经典操作系统主要以提高资源的利用率为主，如内存的管理策略[35] 等。然而，在量子处理器中，由于环境的影响，其量子比特的性能会随时间不断恶化。在此情况下，如果根据固定的单门、双门和读取错误率进行量子程序映射，无法得到有效的结果；只有通过量子操作系统不断检测量子处理器中量子比特的性质，并对量子比特进行自动化校准，才能保证计算的正确性。稳定、良好的量子比特性能是实现多量子程序并行，提高量子资源使用效率的前提[36] 。

综上所述，量子计算机需要特有的量子比特映射、量子并行计算、量子比特校准，当前的经典操作系统不能满足量子计算的需求，开发量子操作系统迫在眉睫。

2.3 本源司南量子操作系统中的基本定义

量子应用

量子应用是指为完成某项或多项计算任务的量子-经典混合程序。

量子任务

量子任务是量子应用发送的基本任务单元。它可以表示为一个量子程序。一个量子应用可以包含多条量子程序，不同量子程序的结果可以通过经典算法进行处理。

量子事务

量子事务是单个量子处理器执行的基本单元。它可包含多个量子任务，每个量子任务作为一个整体要么全部执行，要么全部不执行。

量子线程

量子线程是量子操作系统的基本调度单元，量子操作系统根据量子事务所需资源进行调度。量子事务在获得量子处理器开始执行后称为量子线程, 量子线程是量子事务的执行载体。

量子处理器

量子处理器是量子线程的执行单元，在一段时间内只能执行一个量子线程。在一个量子应用中可使用多个量子处理器。

量子资源

量子资源指遵循量子力学规律，可用于量子信息处理及存储的物理系统，它的基本单位是量子比特。量子资源包括量子处理器、量子存储器等。

3 本源司南框架

3.1 整体架构

本源司南是量子-经典的混合架构，支持包括量子处理器、量子虚拟机和高性能计算集群等多种计算后端。量子计算机的强大计算能力需要经典计算机的辅助（如在解决 NP 问题中，用经典计算机检验量子计算结果是否正确），特别的，量子-经典混合算法本身，如量子机器学习算法、量子化学算法、量子金融算法，在量子计算应用中也起着重要的作用。因此，在完整的量子应用中我们既需要运行量子任务，也需要处理经典信息。本源司南系统服务中，以是否涉及量子计算任务为依据，将服务分为量子服务和经典服务，它们相互配合（参见图 2 本源司南的框架图）。

量子服务负责处理量子任务，并与量子计算后端进行数据交互。它具体包含量子任务调度、量子资源管理、量子比特自动化校准、量子程序编译等，通过以上服务提高多量子程序并行计算的保真度和量子处理器中量子比特的利用率。

经典服务主要负责处理经典计算任务，负责与经典计算后端进行数据交互，同时经典服务也需要兼顾量子计算机环境加载、系统服务与量子设备状态监控等量子服务。具体包括作业调度系统、系统服务监控、系统日志、量子计算机配置、量子设备监控等服务，通过以上服务解决量子-经典混合算法的大规模经典数据处理需求，并兼顾本源司南对量子设备和系统服务的管理，保证系统的稳定性。

量子应用调用以上系统服务，基于量子编程框架和分布式计算框架，可实现量子与经典异构的分布式计算。量子应用可通过系统接口调用经典服务生成量子计算所需输入数据，并根据该数据生成或调用量子程序，然后通过调用接口向量子服务发送量子任务，在得到量子计算结果后还需要通过经典服务对当前结果进行数据分析，生成下一步量子计算所需的输入数据。同时用户也可以通过量子操作系统资源管理器管理量子设备和量子资源。

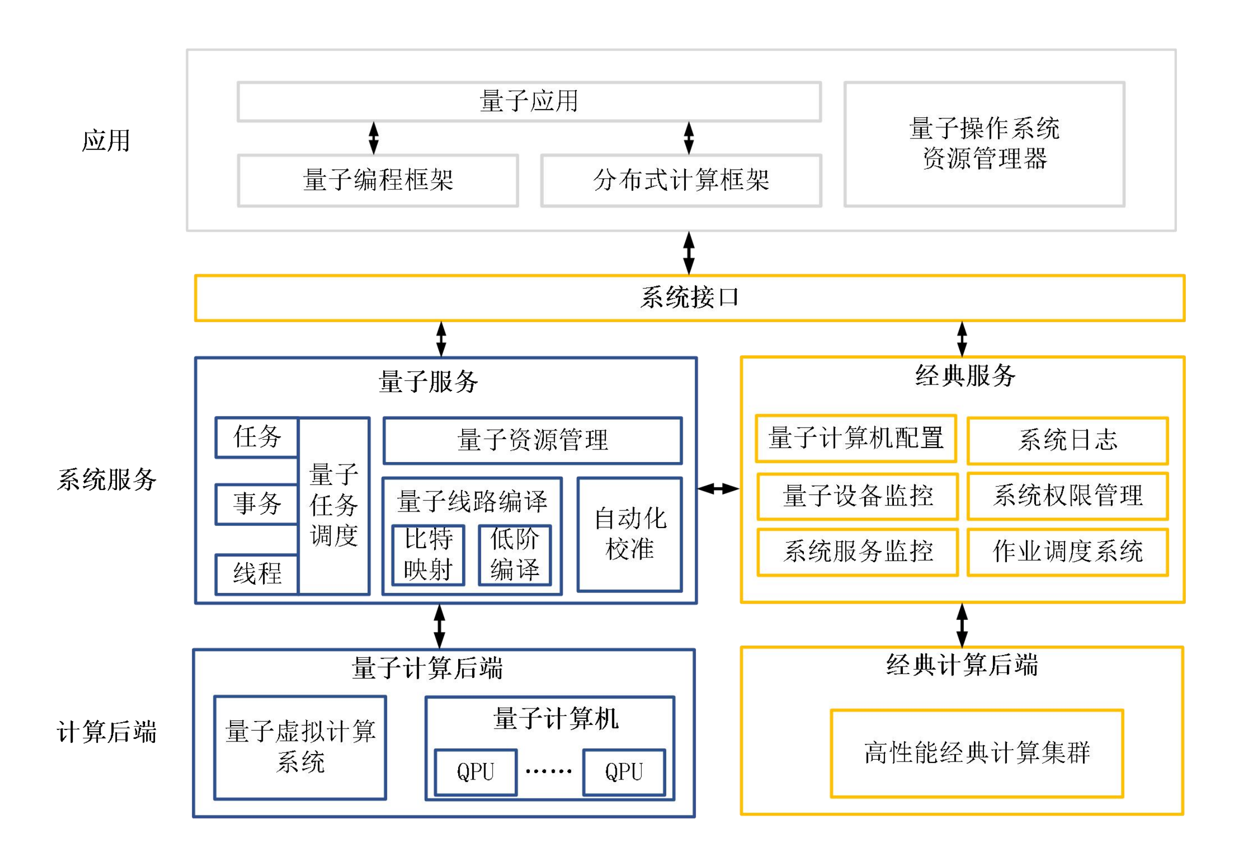


图 2　本源司南框架图

3.2 本源司南的工作流程

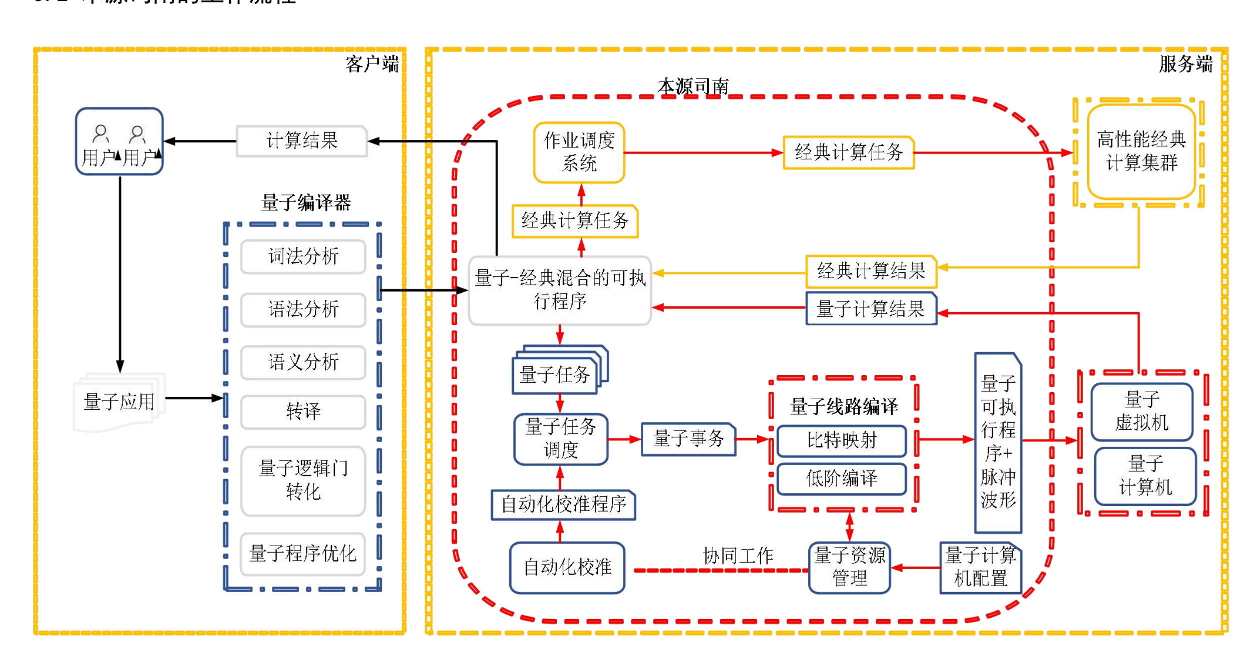


图 3　本源司南工作流程图

本源司南的工作流程如图 3，具体如下：

1.用户通过 QRunes 语言[37] 编写量子-经典混合的量子应用，量子编译器通过词法、语法、语义分析识别量子应用的经典程序部分和量子程序部分，并对量子程序部分进行转译、量子逻辑门转化、量子程序优化等操作，然后将量子应用编译为量子-经典混合的可执行程序，并把它提交到本源司南服务端。如果量子应用有调用高性能经典计算集群的需求，用户在编写量子-经典混合程序时也可使用经典计算的分布式计算

框架编写经典程序。

2.本源司南服务端接收到用户提交的量子-经典混合的可执行程序后，运行该程序，它的经典计算部分可在服务端上位机中运行，如果用户基于分布式计算框架编写了量子-经典混合程序，则在执行经典计算部分时，本源司南会通过作业调度系统把经典计算任务发送给高性能经典计算集群。

3. 量子-经典混合的可执行程序中的量子程序部分会通过任务的形式把量子程序对应的 OriginIR（量子程序中间表示）提交给量子任务调度服务，量子任务调度服务根据当前系统现有的量子任务优先级进行排序，并选取优先级最高且满足量子资源要求的多个量子任务合并成一个量子事务。该量子事务会被提交给量子程序编译服务进行量子比特映射，用于适配量子处理器的拓扑结构，然后经过低阶编译适配目标量子计算机的指令集，最后产生量子可执行程序和脉冲波形。在经过以上一系列操作后，量子事务会被本源司南发送给量子计算设备并等待计算的返回结果。量子事务在被执行之前将会与一个量子线程绑定，该量子线程指定为一个固定的量子处理器提交量子事务，该量子线程会记录量子事务的编号、目标量子处理器编号、量子事务包含的多个量子任务编号、量子事务编译成的量子可执行程序、量子事务占用的量子比特编号。量子事务的计算结果返回时，可通过量子线程回溯到对应的量子任务，并把量子任务的计算结果返回给对应的量子-经典混合的可执行程序。

在运行量子任务的同时，本源司南也会通过自动化校准服务监控并校准量子资源的状态，自动化校准程序会定时检测量子资源中量子比特的状态，如果量子比特状态不满足计算需求，自动化校准服务会通知量子资源管理服务把对应的量子比特设置为待校准状态，并会向量子任务调度服务发送校准任务，本源司南会把校准任务设置为最高优先级，并与其他量子计算任务一起合并为量子事务发送给量子设备执行。

4 本源司南提供的解决方案

我们已经描述了本源司南的总体架构和工作流程，本节将详细阐述本源司南提供的多量子处理器负载均衡、多量子程序并行计算和量子比特自动化校准的解决方案。

4.1 多量子处理器负载均衡

多量子处理器负载均衡是量子任务调度服务需要解决的核心问题。为理解量子任务的调度过程，我们首先介绍量子任务的基本组成元素：

(1) 量子任务所需量子比特数目；

(2) 量子程序中间表示；

(3) 量子处理器编号；

(4) 量子任务类型：分为量子程序和量子比特自动化校准任务两类；

(5) 优先级：量子任务的执行优先级。

如果元素（3）中的量子处理器编号为 A，那么量子任务就只能在 A 上运行，若元素（3）中的编号缺省则由量子任务调度服务根据系统状态自行分配。量子任务调度服务对元素（4）中量子程序和量子比特自动化校准任务执行不同的调度算法。

量子比特自动化校准任务要求响应时间快（需实时响应），具有执行时间短、操作物理量子比特明确等特点。该任务具有最高优先级，优先被执行，为量子计算的可靠性提供保障。而量子程序无需实时响应，且无需事先指定量子处理器及量子比特，可根据多个量子处理器系统可执行区域（详见 4.3 节）的量子比特资源进行高效分配。针对该类型量子任务，量子任务调度服务采取量子高响应比优先的调度算法。它综合考虑了量子任务的要求服务时间（量子编译器对量子任务执行时间进行预估）和等待时间，响应时间比具体定义如下：

$$R_p = \frac{\text{等待时间} + \text{要求服务时间}}{\text{要求服务时间}} = \frac{\text{响应时间}}{\text{要求服务时间}} \qquad (1)$$

R_p 大的量子任务将优先执行。此算法综合了先来先服务和短作业优先两种算法[38] 的优点，引入动态优先权，使量子任务的优先级随着等待时间的增加而提高。它保证了要求服务时间较长的量子任务在等待一段时间后仍会被执行。

依据上述算法，量子任务调度服务的运行过程如图 4 所示。

量子任务调度服务在接收到量子任务后会执行以下步骤：（1）获取该量子任务的要求服务时间，根据 R_p=1，将其并入等待队列；（2）随着此任务等待时间的变化，R_p 也持续变大，并根据所有等待任务的新 R_p 更新等待队列；（3）根据不同量子处理器的可执行区域的大小，调度服务选择优先级位于前列的多个量子任务合并为一个量子事务（该量子事务所需的量子比特数目不大于多个量子处理器中当前最大可执行区域），实现量子事务在多个量子处理器上的合理分配，达到多量子处理器负载均衡；（4）量子事务被量子程序编译服务编译为量子可执行程序和脉冲波形；（5）通过量子线程绑定，在量子处理器上执行计算。

特别强调，量子比特自动化校准任务总是处于等待队列的最前面，它与其它量子程序一起组成量子事务。该量子事务中，量子比特自动化校准任务的量子比特已指定，它不参与量子比特映射；其他量子程序则需要参加量子比特映射以适配量子处理器的拓扑结构。这里对于量子比特自动化校准任务有两点说明：（1）量子处理器分为任务执行区域和校准区域（详见 4.3 节），如果等待队列中有多个自动化校准任务，那么这些任务也仅针对校准区域的量子比特进行校准，不影响任务执行区域的量子任务，所以不会存在导致其他量子程序处于长时间等待而被“饿死”的情况；（2）量子任务调度服务的算法遵循先来先服务的原则，即如果任务队列中有多个量子自动化校准任务，那么它们的优先级也会根据它们到达的先后顺序来确定。

4.2 多量子程序并行计算

目前本源司南实现的是同步并行量子计算：多个量子任务组成一个量子事务后，它们同时开始执行，当一个量子事务完成（量子事务包含的所有量子任务完成）后才能执行下一个量子事务。在执行量子事务前，需要将其逻辑量子比特有效地映射到物理量子比特，以最小化为处理非近邻量子逻辑门而引入的 SWAP 门个数。本源司南同步并行的工作流程如图 4 所示：

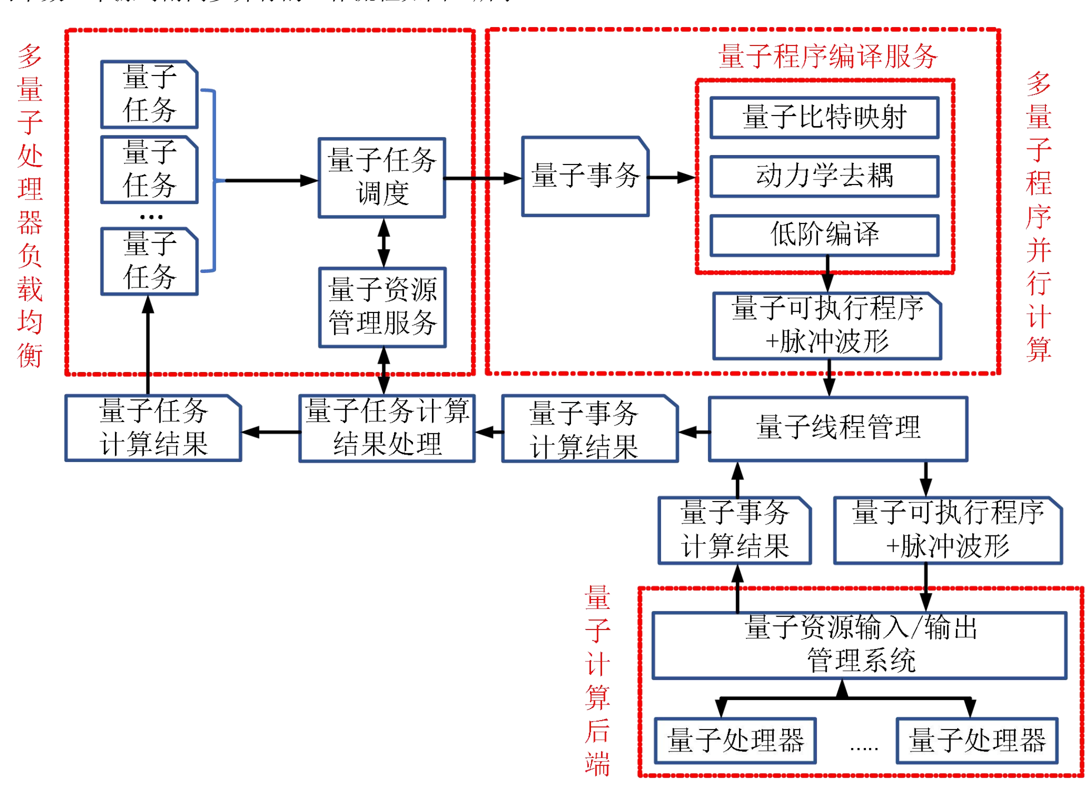


图 4 本源司南多处理器负载均衡及同步并行工作流程图

(1) 本源司南量子比特映射方案

由于量子处理器中量子比特拓扑结构以及量子比特实时状态的不同，需特别设计量子程序的量子比特映射。量子比特映射的核心是最小化量子程序中物理量子比特之间的因非近邻量子门而引入的 SWAP 门的个数。实现每一个非近邻门都需要引入若干 SWAP 门，而有效的量子比特映射算法可以显著减少 SWAP 门

的个数。量子比特映射是一个 token-swap[39] 问题，求其最优解的算法复杂度随比特增长指数增加。

量子比特映射问题是量子计算的一个重要研究方向，目前已提出多种解决方案[40] [41] [42] [43][44] 。量子比特映射问题需考虑两方面的因素：（1）量子比特拓扑结构：在理想的情况下，我们希望所有的两量子比特门都能映射到近邻物理比特上，进而可以直接实现。然而，由于量子比特的拓扑结构限制（如二维超导量子比特仅有 4 个近邻量子比特），在量子程序中将不可避免的出现非近邻的两比特门。而非近邻两量子比特门的实现需要引入大量的 SWAP 门，极大地增加量子线路的逻辑深度。因此量子比特映射的核心任务就是最小化总体的 SWAP 门个数。（2）差异化的比特品质：环境对量子比特的影响不可避免，且每个量子比特受到的影响不同。这导致非近邻逻辑门的不同实现路径，其效果（保真度）不尽相同，因此，我们需寻找保真度最高的那条实现非近邻逻辑门的路径。

本源司南提供的量子比特映射算法同时优化了 SWAP 门的个数和含噪下的非近邻逻辑门保真度。此算法具体实现细节如下：（1）将量子程序转换成有向无环图(Directed acyclic graph, DAG)，其顶点表示为两量子比特量子逻辑门，两个顶点相连表示这两个量子逻辑门有共同的量子比特，方向表示时序；（2）获取 DAG 中入度为 0 的节点，并开始遍历。在遍历过程中，从入度为 0 的顶点中选择可以直接映射（根据量子处理器拓扑结构，不需要加入 SWAP 门就可以直接从逻辑比特映射到物理比特的双门）的顶点，将每种映射记录为一个子图；（3）从 DAG 中剔除已经映射的顶点及与之相连的边，得到新的 DAG。从新的 DAG 中入度为 0 的节点开始进行遍历，在现有子图基础上选择可以直接映射的顶点，以此扩展子图，同时删除无法扩展的子图，不断迭代步骤（3），直到当前入度为 0 的顶点都无法直接映射，则当前剩余的子图即为一个最大子图集合；（4）重复（2）、（3）步骤，直至 DAG 中所有顶点都被遍历，将得到多个最大子图序列;（5）通过 Token-Swapping[45] 算法计算相邻子图的路径消耗，选择最小消耗路径（需要引入的 SWAP 门个数最少），连接各个相邻子图，最终得到多种映射方案；（6）依次计算每种映射方案的整体路径保真度信息，选择最优映射。

本源司南提供的量子比特映射算法通过考虑路径的保真度信息，选择保真度较高的量子比特，从而有效提高了量子程序在真实芯片运行的正确性。算法的伪代码实现如下：

```
Input:   src_QProg (原始量子线路), QPU_adj(物理量子芯片拓扑结构)
Output:  mapped_QC(mapping 后，符合物理量子芯片拓扑结构的量子线路)
将 src_QProg 转换成 DAG;
初始化二维存储结构 sub_graph_vec_2d，用于存放最大子图序列;
// phase_1：遍历 DAG 获取最大子图序列
While    DAG 顶点数目大于 0
         取入度为 0 的顶点 V;
         If(子图序列 S 不为空)
                 If(子图序列 S 的所有子图元素都无法扩展)
                         将当前子图序列 S 记为一个最大子图序列，存入 sub_graph_vec_2d;
                         清空当前子图序列 S;
                         Break;
                 Else
                         根据 QPU_adj 结构扩展子图序列 S，并从 S 中删除无法扩展的子图;
         Else
                 根据 V 在 QPU_adj 结构上的映射可能，初始化子图序列 S;
//phase_2：Token-Swapping，获取最优路径
初始化 best_path_vec，用于存放最优 path;
Foreach S_cur in sub_graph_vec_2d
         通过 Token-Swapping 算法计算 S_cur 中每个子图到 S_next 每个子图的最小 Cost;
         在 sub_best_path_vec 中追加 S_cur 中每个子图到 S_next 的最优路径 best_swap 和对应的 S_next 子图
```

```
                    信息；
        Foreach best_path in best_path _vec
                    计算 best_path 的总体保真度值；
        从 best_path _vec 选择保真度最高的 final_best_path；
        // phase_3：遍历 Final_best_path，生成新的 mapped_QC
        Foreach path_node in Final_best_path
                    If(path_node 是子图信息)
                            将 path_node 转换为量子线路 sub_cir；
                            将 sub_cir 插入 mapped_QC；
                    If(path_node 是 best_swap 信息)
                            在 mapped_QC 中插入 swap-gates；
        return      mapped_QC；
```

4.3 量子比特自动化校准

刻画量子比特质量的性能参数主要包括：相干时间和门操作保真度。其中：

1. 量子比特退相干时间可以描述系统保持量子相干性的时间，同时也可以用来描述量子比特系统与环境耦合的强弱。由于量子算法的实现可能需要大量的门操作，因此量子比特需要保持较长的退相干时间;

2. 量子逻辑门操作是构成量子线路的基本单位，它的误差对量子计算有决定性影响。要实现普适的量子计算，量子逻辑门的误差要小于 1%[46] 。

影响量子比特质量的因素非常复杂，因此，为确保量子比特能够长期有效工作，需要持续实施校准。量子比特校准可分为校验和校准两个阶段。在校验阶段，可使用定期校验、随机巡游等技术，对量子比特的性能参数进行检测，来决定是否需要校准。如需校准，根据误差程度和类型，构成并实施校准。

量子比特的性能参数受多个相互关联的物理参数影响，确定哪些物理参数导致性能参数降低是构成校准的关键步骤，此过程可以由马尔科夫过程表示并实现自动化。我们构建了一个基于部分可观测马尔科夫决策过程的自动化校准决策系统，实现了按需的自动化校准。

除此之外，我们设计了一套实时线上校准方案，利用分块自动化校准等技术，实现了在线校准。该方案根据校验结果将量子处理器动态划分为任务执行区域与校准区域。量子程序编译服务根据所划分的区域，同时编译校准任务与量子任务并组成量子事务发送给量子处理器执行，以此来确保量子任务和校准任务的同时执行，也可确保量子任务运行在量子处理器状态最好的区域上，提升量子任务运行结果表现。

5 实验与结果

为直观体现本源司南操作系统对量子资源的有效管理，我们下面对它进行不同的对比测试。

5.1 相同量子任务运行时间对比

本部分实验均在真实的本源悟源量子处理器上进行。本源悟源量子处理器是本源量子自主研发的 6 比特超导量子处理器夸父 KF C6-130，其保真度、相干时间等各项技术指标均达到国际先进水平。

5.1.1 实验目的及设置

本实验对比接入本源司南前后，多量子处理器（悟源 1 号与悟源 2 号）在处理相同量子任务时的不同效率。通过运行 GHZ 线路的效果来验证本源司南支持单量子处理器并行计算和多量子处理器分布式计算的有效性。

5.1.2 实验步骤

在未接入本源司南的本源量子云平台时，单量子处理器单线路方式下，记录执行 10 次 2 量子比特的 GHZ 线路的运行时间。在接入本源司南的本源量子云平台上，采用单量子处理器多线路并行、双量子处理器单线路、双量子处理器多线路并行等不同方式，分别记录执行 10 次 2 量子比特的 GHZ 线路的运行时间。

将前面的过程重复 10 次。

5.1.3 实验数据及结果

不同方式下，执行实验任务所运行的时间如图 5 所示：

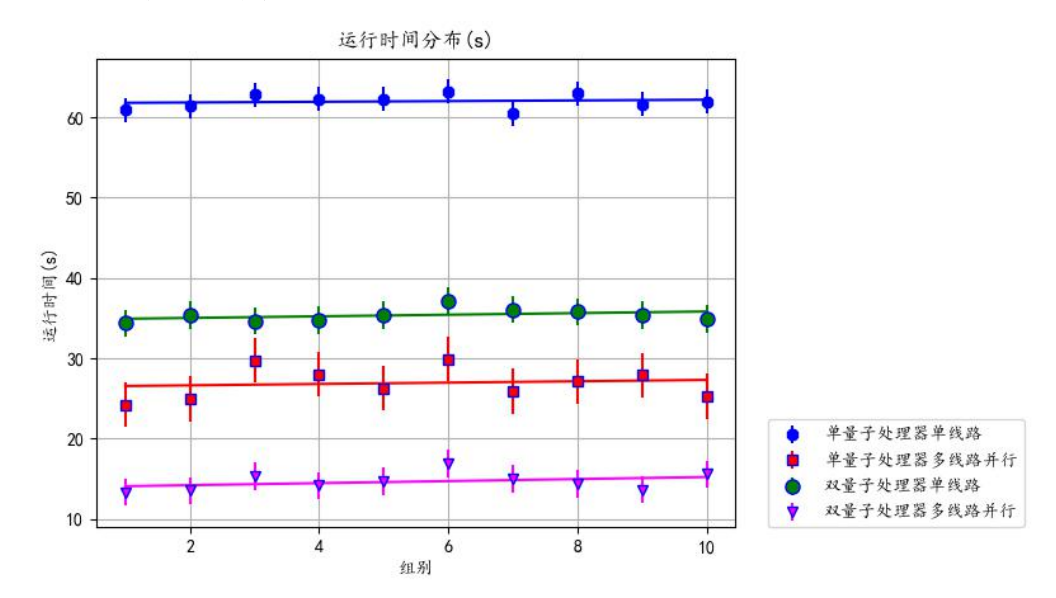


图 5 GHZ 线路的运行时间分布

5.1.4 实验结论

由实验数据可见，本源司南支持单量子处理器并行计算与多量子处理器负载均衡，极大提高量子处理器的使用效率，在运行 2 量子比特的 GHZ 线路中运行效率至少提高 120%。

5.2 量子比特自动化校准效果

本部分实验均在真实的本源悟源量子处理器上进行。

5.2.1 实验目的

为了验证量子比特自动化校准的有效性，为本源悟源设置了一系列保真度阈值条件，当保真度低于阈值时，量子处理器上对应的量子比特将不能再提供服务。分别进行 3 组自动化校准对比实验：单比特量子逻辑门保真度（校准/未校准）对比实验、双比特量子逻辑门保真度（校准/未校准）对比实验和量子处理器（校准/未校准）可运行量子任务数目对比实验。

5.2.2 实验设置及步骤

（1）单比特量子逻辑门保真度（校准/未校准）对比实验设置

1. 两个实验中均每隔为 20 分钟测量一次 X 门的保真度，共测量了 30 次；
2. 在校准实验中校准保真度阈值设置为 0.98。

（2）双比特量子逻辑门保真度（校准/未校准）对比实验设置

1. 两个实验中均每隔为 20 分钟测量一次 CZ 门的保真度，共测量了 30 次；
2. 在校准实验中校准保真度阈值设置为 0.95。

（3）量子处理器（校准/未校准）可运行量子任务数目对比实验设置

1. 统计 20 分钟内量子处理器重复运行了同一个量子任务（两比特 GHZ）的次数，共统计了 30 个周期（每个周期 20 分钟）。

5.2.3 实验数据及结果

(1) 单比特量子逻辑门保真度（校准 VS 未校准）对比实验数据

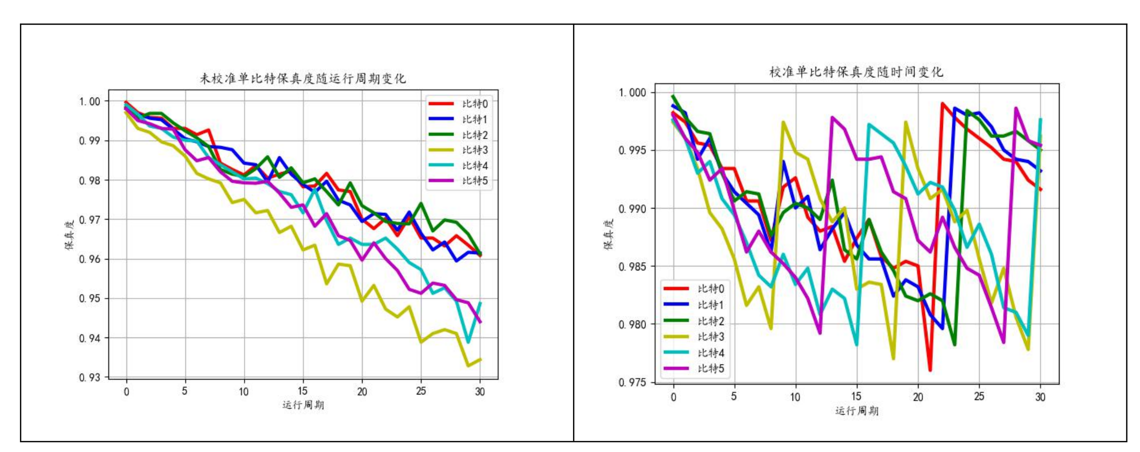


图 6 单比特保真度（校准 VS 未校准）变化比对图

(2) 双比特量子逻辑门保真度（校准VS未校准）对比实验数据

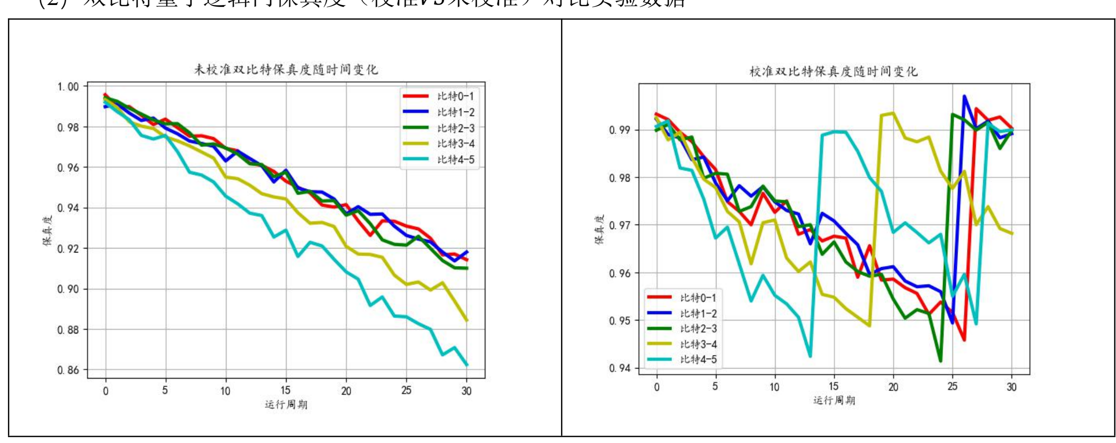


图 7 双比特保真度（校准VS未校准）变化对比图

(3) 量子处理器（校准 VS 未校准）可运行量子任务数对比实验数据

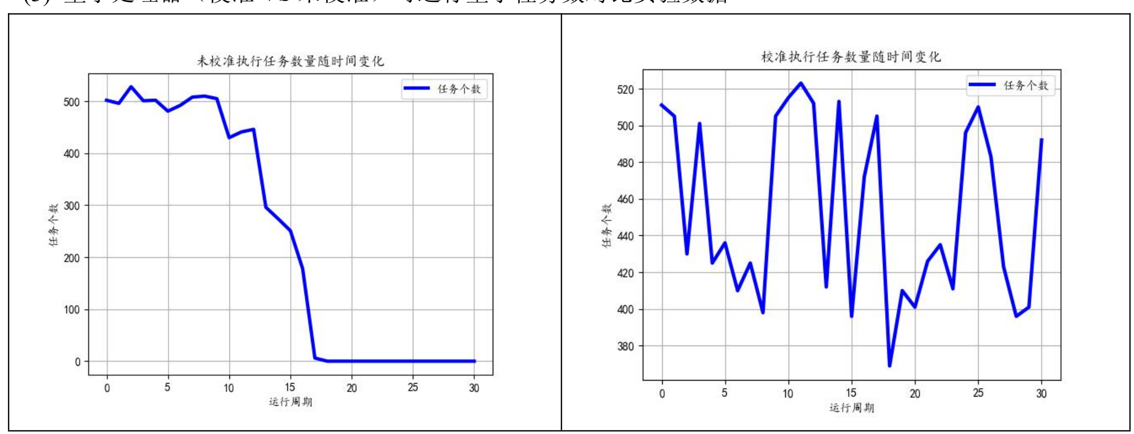


图 8 量子计算机（校准 VS 未校准）可运行量子任务数比对图

5.2.4 实验结论

从上述 3 组实验可以看出量子处理器在未开启自动化校准服务时单比特量子逻辑门保真度和双比特量子逻辑门保真度呈单一下降趋势；在接入自动化校准服务后单比特量子逻辑门和双比特量子逻辑门保真度可保持在 0.98 和 0.95 以上。开启自动化校准服务后的量子处理器可持续提供服务，其运行程序的数目远大于未开启服务的量子处理器，后者在运行一段时间后就无法再提供服务。

5.3 量子比特映射效果

5.3.1 实验目的

为了验证本源司南对量子程序映射适配超导量子处理器拓扑结构和保真度的有效性，对 4 种代表性的量子程序进行了映射优化前后的保真度对比。

5.3.2 实验设置

本实验使用本源量子含噪声虚拟机完成，在虚拟机中构建的量子处理器的拓扑结构和量子比特间保真度数据如图 9：

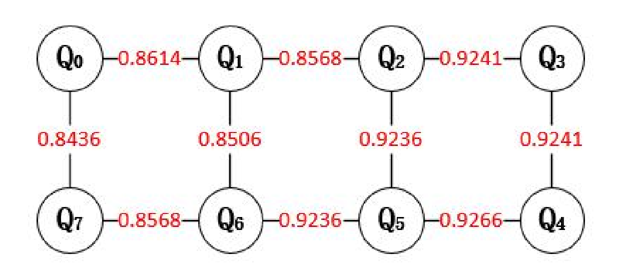


图 9 本源量子含噪声虚拟机中量子处理器的拓扑结构与量子比特间双门保真度

5.3.3 实验步骤

1. 构建拓扑结构，模拟芯片为 2 × 4 的无向图拓扑结构，相邻比特间均可应用双门。
2. 基于 QPanda 含噪声虚拟机进行配置（退相干、去极化噪声），在这里只考虑双门 CZ 的影响。
3. 构建实验量子线路，运行 QFT（Quantum Fourier Transform）、GHZ（Greenberger–Horne–Zeilinger experiment）、DJ（Deutsch–Jozsa algorithm）、BV（Bernstein–Vazirani algorithm）算法线路。
4. 以上 4 个线路经过 BMT 和本源司南量子比特映射优化，分别得到两种不同映射线路。
5. 将转化后的不同映射线路分别进行 QST（Quantum State Tomography）计算，得到对应保真度，以判定量子比特映射的效果。

5.3.4 实验数据及结果

(1) QFT 线路

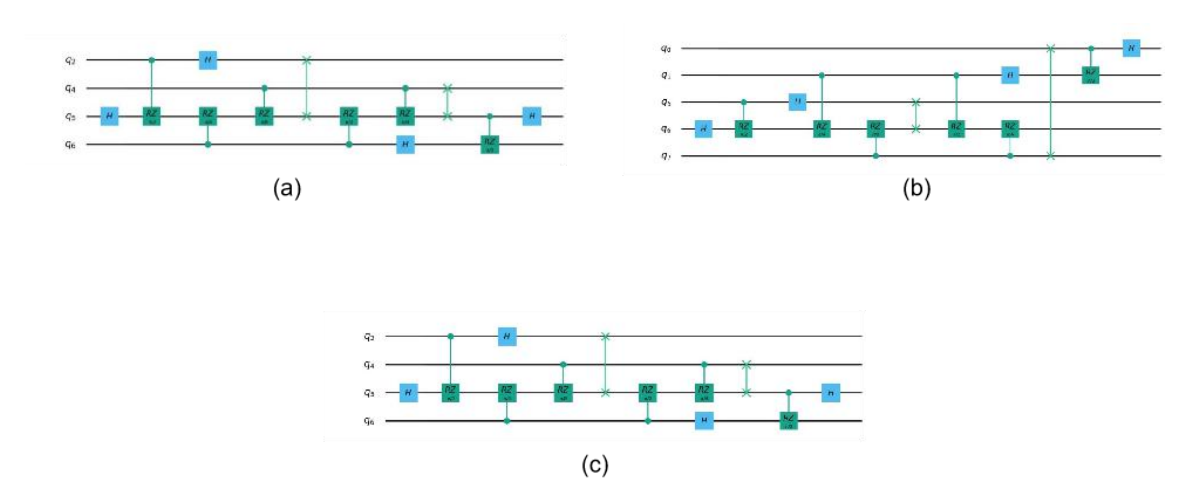


图 10 (a)线路表示 QFT 原始线路；(b)是 BMT 算法映射的 QFT 线路，在这里量子比特映射为 0 -> 5、1 -> 6、2 -> 1、3 -> 0；(c)是经过本源司南映射的 QFT 线路，在这里量子比特映射为 0 -> 2、1 -> 4、2 -> 6、3 -> 5。

(2) GHZ 线路

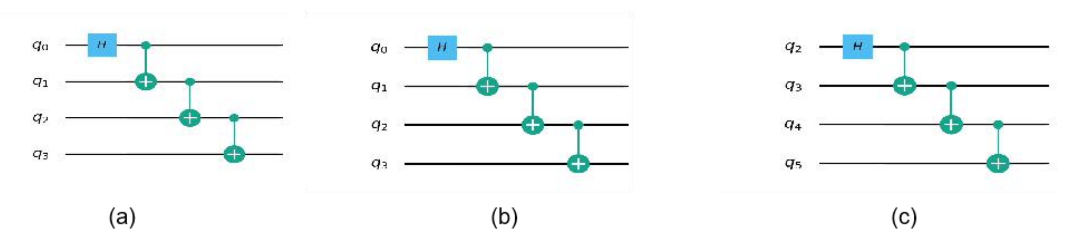


图 11 (a)线路表示 GHZ 原始线路；(b)是 BMT 算法映射的 GHZ 线路，在这里量子比特映射为 0 -> 0、1 -> 1、2 -> 2、3 -> 3；(c)是经过本源司南映射的 GHZ 线路，在这里量子比特映射为 0 -> 2、1 -> 3、2 -> 4、3 -> 5。

(3) DJ 线路

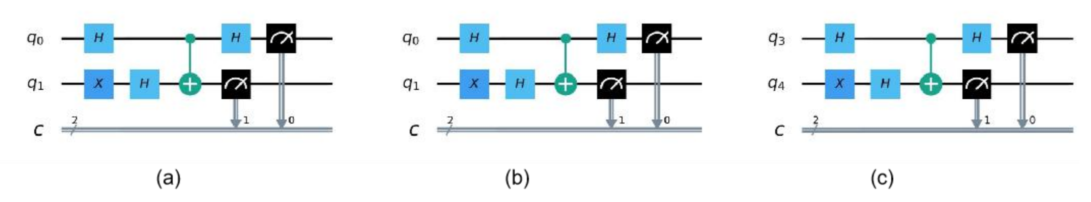


图 12 (a)线路表示 DJ 原始线路; (b)是 BMT 算法映射的 DJ 线路，在这里量子比特映射为 0 -> 0、1 -> 1，线路未变化;((c)是经过本源司南映射的 DJ 线路，在这里量子比特映射为 0 -> 3、1 -> 4。

(4) BV 线路

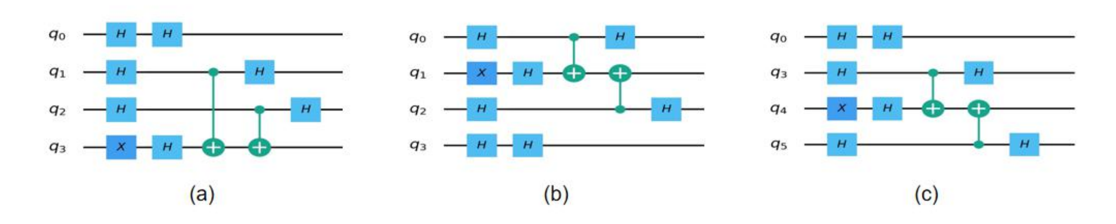


图 13 (a)线路表示 BV 原始线路；(b)是 BMT 算法映射的 BV 线路，在这里量子比特映射为 0 -> 3、1 -> 0、2 -> 2、3 -> 1；(c)是经过本源司南映射的 BV 线路，在这里量子比特映射为 0 -> 0、1 -> 3、2 -> 5、3 -> 4

(5) 量子线路保真度结果对比（误差主要来自于统计误差）

表 1　4 种量子线路的保真度结果对比

	QFT	GHZ	DJ	BV
BMT 算法	0.1266±0.004	0.4305±0.005	0.4305±0.004	0.5454±0.005
本源司南映射优化	0.2139±0.003	0.6032±0.004	0.8596±0.004	0.7372±0.004

5.3.5 实验结论

针对以上 4 种代表性的量子程序的线路，本源司南均比 BMT 算法起到更好的映射效果，映射后线路的保真度至少提升 10%。

6 未来趋势与展望

本源司南的宗旨是高效利用珍贵的量子计算资源，并充分与经典计算相结合，它提供的量子任务调度、量子资源管理、量子程序编译、量子比特自动化校准等服务可以达到高效利用和管理量子计算资源的目的。本源司南未来将支持基于量子线路拆分的用户无感知量子分布式计算、量子异步并行计算，适配不同物理体系的量子处理器，进一步提升量子比特自动化校准、量子程序编译、量子任务调度的效率。支持多物理体系量子分布式计算、高性能经典计算机集群加量子处理器混合计算和量子处理器加量子虚拟机混合计算。

References:

[1] Nielsen M A , Chuang I L . Quantum Computation and Quantum Information[J]. Mathematical Structures in Computer Science, 2002, 17(6):1115-1115.

[2] Grover L.K, A fast quantum mechanical algorithm for database search, Proceedings, 28th Annual ACM Symposium on the Theory of Computing, (May 1996) p. 212.

[3] A. W. Harrow, A. Hassidim, and S. Lloyd, "Quantum algorithm for linear systems of equations," Phys. Rev. Lett. 103.15 (2009), p. 150502.

[4] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," SIAM review, vol. 41, no. 2, pp. 303–332, 1999.

[5] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, "Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets," Nature, vol.549, no. 7671, pp. 242–246, 2017.

[6] Ghosh B, Kozarević E. Identifying explosive behavioral trace in the CNX nifty index: A quantum finance approach[J]. Investment Management & Financial Innovations, 2018, 15(1): 208.

[7] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, "Quantum machine learning," Nature, vol. 549, no. 7671, pp.195–202, 2017.

[8] Schuld M, Sinayskiy I, Petruccione F. An introduction to quantum machine learning[J]. Contemporary Physics, 2015, 56(2): 172-185.

[9] Houck, A., Türeci, H. & Koch, J. On-chip quantum simulation with superconducting circuits. Nature Phys 8, 292–299 (2012). https://doi.org/10.1038/nphys2251.

[10] Li, Xiaoqin & Wu, Yanwen & Steel, Duncan & Gammon, D & Stievater, Todd & Katzer, D & Park, D. & Piermarocchi, Carlo & Sham, L. (2003). An All-Optical Quantum Gate in a Semiconductor Quantum Dot. Science (New York, N.Y.). 301. 809-11. 10.1126/science.1083800.

[11] H. Häffner, C.F. Roos, R. Blatt, Quantum computing with trapped ions, Physics Reports, Volume 469, Issue 4, 2008, Pages 155-203, ISSN 0370-1573.

[12] Kielpinski, D., Monroe, C. & Wineland, D. Architecture for a large-scale ion-trap quantum computer. Nature 417, 709–711 (2002).

[13] Lvovsky, A., Sanders, B. & Tittel, W. Optical quantum memory. Nature Photon 3, 706–714 (2009).

[14] O'Brien JL. Optical quantum computing. Science. 2007 Dec 7;318(5856):1567-70.

[15] Guo Y, Zhang YF, Bao XY, Han TZ, Tang Z, Zhang LX, Zhu WG, Wang EG, Niu Q, Qiu ZQ, Jia JF, Zhao ZX, Xue QK. Superconductivity modulated by quantum size effects. Science. 2004 Dec 10;306(5703):1915-7.

[16] Suter, D. and Gonzalo A. Alvarez. "Colloquium : Protecting quantum information against environmental noise." Reviews of Modern Physics 88 (2016): 041001..

[17] Wu R B, Chu B, Owens D H, et al. Data-driven gradient algorithm for high-precision quantum control[J]. Physical Review A, 2018, 97(4): 042122.

[18] Botero U J, Tehranipoor M M, Forte D. Upgrade/downgrade: Efficient and secure legacy electronic system replacement[J]. IEEE Design & Test, 2018, 36(1): 14-22.

[19] Orgiazzi J L , Deng C , Layden D , et al. Flux qubits in a planar circuit quantum electrodynamics architecture: quantum control and decoherence[J]. Phys.rev.b, 2016, 93(10):104518.

[20] LaRose R. Overview and comparison of gate level quantum software platforms[J]. Quantum, 2019, 3: 130.

[21] Alexeev Y, Bacon D, Brown K R, et al. Quantum computer systems for scientific discovery[J]. PRX Quantum, 2021, 2(1): 017001.

[22] Kottmann J S, Alperin-Lea S, Tamayo-Mendoza T, et al. Tequila: A platform for rapid development of quantum algorithms[J]. Quantum Science and Technology, 2021, 6(2): 024009.

[23] Arute, F., Arya, K., Babbush, R. et al. Quantum supremacy using a programmable superconducting processor. Nature 574, 505–510 (2019). https://doi.org/10.1038/s41586-019-1666-5.

[24] Harrow, A., Montanaro, A. Quantum computational supremacy. Nature 549, 203–209 (2017).

[25] Vasileska D, Goodnick S M, Klimeck G. Computational Electronics: semiclassical and quantum device modeling and simulation[M]. CRC press, 2017.

[26] Ajagekar A, Humble T, You F. Quantum computing based hybrid solution strategies for large-scale discrete-continuous optimization problems[J]. Computers & Chemical Engineering, 2020, 132: 106630.

[27] Henry Corrigan-Gibbs, David J. Wu, and Dan Boneh. Quantum Operating Systems[J]. HotOS '17: Proceedings of the 16th Workshop on Hot Topics in Operating Systems.

[28] Honan, R., Lewis, T.W., Anderson, S. and Cooke, J., 2020, October. A Quantum Computer Operating System. In International Conference on Algorithms and Architectures for Parallel Processing (pp. 415-431). Springer, Cham.

[29] https://www.riverlane.com/news/2020/12/riverlane-release-first-version-of-quantum-operating-system-deltaflow/.

[30] E. Lucero, M. Hofheinz, M. Ansmann, R. C. Bialczak, N. Katz, M. Neeley, A. D. O'Connell, H. Wang, A. N. Cleland, and J. M. Martinis. High-fidelity gates in a single Josephson qubit. Phys. Rev. Lett., 100:247001, Jun 2008.

[31] Richard Jozsa (1994) Fidelity for Mixed Quantum States, Journal of Modern Optics, 41:12, 2315-2323.

[32] Leonardo Banchi, Samuel L. Braunstein, and Stefano Pirandola, Quantum Fidelity for Arbitrary Gaussian States, Phys. Rev. Lett. 115, 260501.

[33] Inverso O, Trubiani C. Parallel and distributed bounded model checking of multi-threaded programs[C]//Proceedings of the 25th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming. 2020: 202-216.

[34] Asyabi E, Sharafzadeh E, SanaeeKohroudi S A, et al. CTS: An operating system CPU scheduler to mitigate tail latency for latency-sensitive multi-threaded applications[J]. Journal of Parallel and Distributed Computing, 2019, 133: 232-243.

[35] Chen J, Du C, Xie F, et al. Scheduling non-preemptive tasks with strict periods in multi-core real-time systems[J]. Journal of Systems Architecture, 2018, 90: 72-84.

[36] Hartmut Neven, John Martinis, et al. Physical qubit calibration on a directed acyclic graph. arXiv:1803.03226v1 [quant-ph].

[37] Zhao-Yun Chen, Guo-Ping Guo. QRunes: High-Level Language for Quantum-Classical Hybrid Programming. arXiv:1901.08340[quant-ph].

[38] Andrew S. Tanenbaum and Herbert Bos. 2014. Modern Operating Systems (4th. ed.). Prentice Hall Press, USA.

[39] Jun Kawahara, Toshiki Saitoh, and Ryo Yoshinaka. The time complexity of the token swapping problem and its parallel variants. In International Workshop on Algorithms and Computation, pages 448–459. Springer, 2017.

[40] Gushu Li, Yufei Ding, and Yuan Xie. Tackling the qubit mapping problem for NISQ-era quantum devices. In 24th ASPLOS, pages 1001–1014. ACM, 2019.

[41] Prakash Murali, Jonathan M Baker, Ali Javadi Abhari, Frederic T Chong, and Margaret Martonosi. Noise-adaptive compiler mappings for noisy intermediatescale quantum computers. In 24th ASPLOS, pages 1015–1029, Providence, RI, USA, 2019. ACM.

[42] Robert Wille, Lukas Burgholzer, and Alwin Zulehner. Mapping quantum circuits to IBM QX architectures using the minimal number of SWAP and H operations. In 56th DAC, page 142. ACM, 2019.

[43] Haowei Deng, Yu Zhang, and Quanxi Li. 2020. Codar: a contextual duration-aware qubit mapping for various NISQ devices. In Proceedings of the 57th ACM/EDAC/IEEE Design Automation Conference (DAC '20). IEEE Press, Article 208, 1–6.

[44] Bin-Han Lu, Yu-Chun Wu, Wei-Cheng Kong, Qi Zhou, Guo-Ping Guo. Special-Purpose Quantum Processor Design. arXiv:2102.01228 [quant-ph].

[45] Bonnet, É., Miltzow, T. & Rzążewski, P. Complexity of Token Swapping and Its Variants. Algorithmica 80, 2656–2682 (2018).

[46] Kelly, J., Barends, R., Fowler, A. et al. State preservation by repetitive error detection in a superconducting quantum circuit. Nature 519, 66–69 (2015).